```
Attribute VB_Name = "Engine"
    Option Explicit



'.....................................................................
Sub ConvertEq5andEq6(ByVal z1 As Integer, ByVal z2 As Integer)

'z1 is the old yarn type
'z2 is the new yarn type

    If z1 = z2 Then Exit Sub

    Select Case z1          'convert to combed ring
      Case 1
        Eq(5) = Eq(5) / 1.04    'from carded ring
        Eq(6) = Eq(6) / 0.96
      Case 2
        Eq(5) = Eq(5) / 0.9     'from carded rotor
        Eq(6) = Eq(6) / 0.9
      Case 3
        Eq(5) = Eq(5) / 1.2     'from two-fold
        Eq(6) = Eq(6) / 0.93
    End Select

    Select Case z2       'convert to new yarn type
      Case 1
        Eq(5) = Eq(5) * 1.04    'to carded ring
        Eq(6) = Eq(6) * 0.96
      Case 2
        Eq(5) = Eq(5) * 0.9     'to carded rotor
        Eq(6) = Eq(6) * 0.9
      Case 3
        Eq(5) = Eq(5) * 1.2     'to two-fold
        Eq(6) = Eq(6) * 0.93
    End Select


End Sub




Public Sub GetDataSet(Inputs() As Single, EngineEr As Boolean)
'This is the Starfish Engine

'Stop
      'Inputs(1) is first target
      'Inputs(2) is second target
      'Inputs(3) is yarn count 1
```

```
      'Inputs(4) is SL 1
'      'Inputs(5) is inlay count when FabTyp = 6
      'Inputs(5) is inlay count when FabTyp = Fab2tf
      'Inputs(6) is SL 2
      'Inputs(7) is number of Needles
      'Inputs(8) is Targets Unit for this calculation
      'Inputs(9) is Run-in Ratio
      'EngineEr is the error flag

    Dim Needles As Integer, RunRat As Single
    Dim TempTarUnit As Integer
    Dim x1 As Single, x2 As Single, x3 As Single
    Dim RefCrses As Single, RefWales As Single
    Dim RefWt As Single, RefWid As Single
    Dim RefYld As Single
    Dim RefTex As Single, RefSL As Single
    Dim InRefTex As Single, InRefSL As Single
    Dim EffRefTexW As Single, EffRefTexC As Single
    Dim KTRefTex As Single, KTRefSL As Single
    Dim Msg, Title, Response

    On Error GoTo ErrorHandler

    Response = vbNo
    Needles = Inputs(7)
    RunRat = Inputs(9)
    TempTarUnit = TarUnit    'save actual TarUnit

    P(23) = Inputs(3)        'yarn count as knitted, user units

    'tex as knitted
    If CountSys = 0 Then P(1) = Inputs(3)
    If CountSys = 1 Then P(1) = 590.54 / Inputs(3)
    If CountSys = 2 Then P(1) = 1000 / Inputs(3)
    RefTex = Eq(1) * P(1)
    P(2) = RefTex

    'ref count, user units
    If CountSys = 0 Then P(15) = RefTex
    If CountSys = 1 Then P(15) = 590.54 / RefTex
    If CountSys = 2 Then P(15) = 1000 / RefTex

    'SL, CL & TF as knitted
    P(24) = Inputs(4)                          'SL user
    P(3) = ConvertSL(Inputs(4), TiUnit, SLcm)   'SL cm
    'CL user
    If TiUnit = SLin Then
      P(25) = P(24) * Needles
    Else
      P(25) = P(3) * Needles
    End If
    P(5) = Sqr(P(1)) / P(3)     'TF as knitted
```

```
    'adjust CL for 1x1 rib
    If FabTyp = FabRib1 Then P(25) = P(25) * 2
    'adjust CL for 2x2 rib
    If FabTyp = FabRib2 Then P(25) = P(25) * 2

    RefSL = Eq(2) * P(3)
    P(4) = RefSL
    P(6) = Sqr(P(2)) / RefSL

    'ref SL, user units
    P(16) = ConvertSL(RefSL, SLcm, TiUnit)

    If FabTyp = Fab2tf Then      'two thread fleece
      P(32) = Inputs(5)     'inlay count as knitted, user units
      'inlay count as knitted, tex
      If CountSys = 0 Then P(26) = Inputs(5)
      If CountSys = 1 Then P(26) = 590.54 / Inputs(5)
      If CountSys = 2 Then P(26) = 1000 / Inputs(5)
      'Eff Tex as knitted is used only to calculate Eff TF
      'which currently is not used
      P(35) = P(1) + 0.125 * P(26)
      P(28) = Sqr(P(35)) / P(3)

      'inlay SL & TF & CL, as knitted default
      P(27) = ConvertSL(Inputs(6), TiUnit, SLcm)    'SL cm
      P(33) = Inputs(6)                             'SL user
      'CL user
      If TiUnit = SLin Then
        P(34) = Inputs(6) * Needles
      Else
        P(34) = P(27) * Needles
      End If

      InRefTex = Eq(10) * P(26)
      InRefSL = Eq(11) * P(27)
      EffRefTexW = RefTex + 0.3288 * InRefTex
      EffRefTexC = RefTex - 0.0896 * InRefTex
      'temporary fix to prevent EffRefTexC going negative
      If EffRefTexC < 1 Then
        GoTo ErrorHandler
'        EffRefTexC = 1
'        MsgBox "Unrealistic fabric quality" & NewLine & "Disregard these
results" & NewLine & "and change Count values"
      End If
      P(29) = InRefTex
      P(30) = InRefSL
      P(36) = EffRefTexW
      P(39) = EffRefTexC
      P(31) = Sqr(P(36)) / P(4)
      'ref inlay count in user units
      If CountSys = 0 Then P(37) = InRefTex
```

```
      If CountSys = 1 Then P(37) = 590.54 / InRefTex
      If CountSys = 2 Then P(37) = 1000 / InRefTex
      'ref inlay SL in user units
      P(38) = ConvertSL(InRefSL, SLcm, TiUnit)
    End If

    If FabTyp = FabSxt Or FabTyp = FabXt6 Then   'sxt or xt6
      P(54) = RunRat
      P(40) = P(1)                    'KT count as knitted tex
      P(41) = P(3) * RunRat           'KT SL as knitted, cm
      P(42) = Sqr(P(40)) / P(41)
      KTRefTex = Eq(12) * P(40)
      KTRefSL = Eq(13) * P(41)
      P(43) = KTRefTex                'KT fin ref tex
      P(44) = KTRefSL
      P(45) = Sqr(P(43)) / P(44)
      'KT count as knitted user (same as all-knit)
      If CountSys = 0 Then P(46) = P(1)
      If CountSys = 1 Then P(46) = 590.54 / P(1)
      If CountSys = 2 Then P(46) = 1000 / P(1)
      'ref KT count, tex
      If CountSys = 0 Then P(49) = KTRefTex
      If CountSys = 1 Then P(49) = 590.54 / KTRefTex
      If CountSys = 2 Then P(49) = 1000 / KTRefTex

      'K&T SL & CL in user units
      P(47) = ConvertSL(P(41), SLcm, TiUnit)    'K&T SL as knitted, user
      P(50) = ConvertSL(KTRefSL, SLcm, TiUnit)  'K&T SL ref, user
      If TiUnit = SLin Then
         P(48) = P(47) * Needles                'K& T CL, user
      Else
         P(48) = P(41) * Needles
      End If
    End If

    Select Case FabState
      Case 1    'finished
        TarUnit = Inputs(8)
        GoSub FillParray
      Case 2    'reference
        TarUnit = 0                  'targets are shrinkage
        Inputs(1) = 0
        Inputs(2) = 0
        GoSub FillParray
    End Select
    TarUnit = TempTarUnit       'restore original targets

    On Error GoTo 0

If Not EngineEr Then Exit Sub

ErrorHandler:
```

```
    EngineEr = True
    Msg = "STARFISH has detected an incompatibility in your data" & NewLine
    Msg = Msg & "which could lead to a fatal error in the calculations" &
NewLines2
    Msg = Msg & "It may be possible to solve the problem by" & NewLine
    Msg = Msg & "re-setting your Model to Default Status    " & NewLine
    Msg = Msg & "To try this, click on OK" & NewLines2
    Msg = Msg & "If the error persists then the most likely reason" & NewLine
    Msg = Msg & "is that your DESkey DK3 is not present or" & NewLine
    Msg = Msg & "is not properly located in a free USB port" & NewLines2
    Msg = Msg & "To clear the problem, insert your DESkey DK3 and click on OK" &
NewLines2
    Msg = Msg & "Otherwise, please make a careful written description of " &
NewLine
    Msg = Msg & "the conditions that are causing the error to arise" & NewLine
    Msg = Msg & "and contact your STARFISH supplier" & NewLines2
    Msg = Msg & "Press Cancel to Exit the program" & NewLines2
    Title = " Missing or Incompatible Data"
    Response = MsgBox(Msg, BtnOKCancel, Title)
    If Response = vbCancel Then
      'True indicates error condition
      DoCloseDown True
    End If
   lHandle = FindDESkey(&HBB006B, vbNullString)


Exit Sub


FillParray:
Rem ------------------------------------ FILL THE P ARRAY

   'assign finishing targets
    Select Case TarUnit
      Case 0    'LShr & WShr
        P(21) = Inputs(1)
        P(22) = Inputs(2)
      Case 1    'Crses & Wales
        P(17) = Inputs(1)
        P(18) = Inputs(2)
        GoSub FinCoursesDefault
        GoSub FinWalesDefault
      Case 2    'Wt & Wid
        P(19) = Inputs(1)
        P(20) = Inputs(2)
        GoSub FinWidthDefault
        GoSub FinWeightDefault
      Case 3    'Wt & Crses
        P(19) = Inputs(1)
        P(17) = Inputs(2)
        GoSub FinWeightDefault
        GoSub FinCoursesDefault
```

```
      Case 4    'LShr & Wid
        P(21) = Inputs(1)
        P(20) = Inputs(2)
        GoSub FinWidthDefault
      Case 5    'Wt & LShr
        P(19) = Inputs(1)
        P(21) = Inputs(2)
        GoSub FinWeightDefault
      Case 6    'Crses & Wid
        P(17) = Inputs(1)
        P(20) = Inputs(2)
        GoSub FinWidthDefault
        GoSub FinCoursesDefault
    End Select
    '.........................calculate fin ref dimensions
    'Reference Courses
    Select Case FabTyp
'      Case 0, 1, 2, 3
      Case FabInt To FabDxt
        x1 = RefTex
        x2 = RefSL
'      Case 4
      Case FabSxt
        x1 = (RefTex + KTRefTex) / 2
        x2 = (0.85 * RefSL) + (0.15 * KTRefSL)
'      Case 5
      Case FabXt6
        x1 = (RefTex + 2 * KTRefTex) / 3
        x2 = (0.95 * RefSL) + (0.05 * KTRefSL)
'      Case 6
      Case Fab2tf
        x1 = EffRefTexC
        x2 = RefSL
    End Select
    RefCrses = Sc / x2 - Fc * x1 ^ -0.7
    P(7) = RefCrses

    'Reference Wales
    Select Case FabTyp
'      Case 0, 1, 2, 3
      Case FabInt To FabDxt
        x1 = RefTex
        x2 = RefSL
'      Case 4
      Case FabSxt
        x1 = (RefTex + KTRefTex) / 2
        x2 = (0.15 * RefSL) + (0.85 * KTRefSL)
'      Case 5
      Case FabXt6
        x1 = (RefTex + 2 * KTRefTex) / 3
        x2 = (0.35 * RefSL) + (0.65 * KTRefSL)
'      Case 6
```

```
      Case Fab2tf
        x1 = EffRefTexW
        x2 = RefSL
    End Select
    RefWales = Sw / x2 + Fw * x1 ^ -0.61
    P(8) = RefWales

    'Reference Weight
    Select Case FabTyp
'      Case 0, 1
      Case FabInt, FabRib1, FabRib2
        RefWt = RefTex * RefSL * RefWales * RefCrses * 0.2
'      Case 2, 3
      Case FabPsj, FabDxt
        RefWt = RefTex * RefSL * RefWales * RefCrses * 0.1
'      Case 4
      Case FabSxt
        x1 = RefTex * RefSL * RefWales * RefCrses / 2 * 0.1
        x2 = KTRefTex * KTRefSL * RefWales * RefCrses / 2 * 0.1
        RefWt = x1 + x2
'      Case 5
      Case FabXt6
        x1 = RefTex * RefSL * RefWales * RefCrses / 3 * 0.1
        x2 = KTRefTex * KTRefSL * RefWales * RefCrses * 2 / 3 * 0.1
        RefWt = x1 + x2
'      Case 6
      Case Fab2tf
        x1 = RefTex * RefSL * RefWales * RefCrses * 0.1
        x2 = InRefTex * InRefSL * RefWales * RefCrses * 0.1
        RefWt = x1 + x2
    End Select
    P(9) = RefWt

    'Reference Width
    RefWid = Needles / RefWales
    P(10) = RefWid

    'Reference Yield
    'P(9) is the reference weight in gsm
    'P(10) is the reference open width in cm
    x1 = P(10) / 100     'width in metres
    x2 = x1 * MTRconverter     'width in yards
    Select Case YldUnit             'convert to
      Case 0
        RefYld = P(9) * x1                  'g/m
      Case 1
        RefYld = P(9) * GSMconverter * x2          'o/y
      Case 2
        RefYld = P(9) * x1 * 0.9144         'g/y
      Case 3
        RefYld = 1000 / P(9) / x1           'm/kg
      Case 4
```

```
    RefYld = 33.91 / P(9) / x2 * 16     'y/lb
  Case 5
    RefYld = 1000 / P(9)                'sm/kg
  Case 6
    RefYld = 33.91 / P(9) * 16          'sy/lb
End Select
P(51) = RefYld

'.............................convert ref to as del, default units
Select Case TarUnit
  Case 0                                'L & W Shrinkage
    P(11) = P(7) * (100 - P(21)) / 100
    P(12) = P(8) * (100 - P(22)) / 100
    P(13) = P(9) * (100 - P(22)) * (100 - P(21)) / 10000
    P(14) = P(10) / (100 - P(22)) * 100
  Case 1                                ' fixed courses & wales
    P(21) = 100 * (P(7) - P(11)) / P(7)
    P(22) = 100 * (P(8) - P(12)) / P(8)
    P(13) = P(9) * (100 - P(22)) * (100 - P(21)) / 10000
    P(14) = RefWid * 100 / (100 - P(22))
  Case 2                                ' fixed weight & width
    P(22) = 100 * (P(14) - P(10)) / P(14)
    P(21) = 100 - 10000 / (100 - P(22)) * P(13) / P(9)
    P(11) = P(7) * (100 - P(21)) / 100
    P(12) = P(8) * (100 - P(22)) / 100
  Case 3                                ' fixed weight & courses
    P(21) = 100 * (P(7) - P(11)) / P(7)
    P(22) = 100 - 10000 / (100 - P(21)) * P(13) / P(9)
    P(12) = P(8) * (100 - P(22)) / 100
    P(14) = RefWid * 100 / (100 - P(22))
  Case 4                                ' fixed % LS & Width
    P(11) = P(7) * (100 - P(21)) / 100
    P(22) = 100 * (P(14) - P(10)) / P(14)
    P(12) = P(8) * (100 - P(22)) / 100
    P(13) = P(9) * (100 - P(22)) * (100 - P(21)) / 10000
  Case 5                                ' fixed weight & LS
    P(22) = 100 - 10000 / (100 - P(21)) * P(13) / P(9)
    P(11) = P(7) * (100 - P(21)) / 100
    P(12) = P(8) * (100 - P(22)) / 100
    P(14) = RefWid * 100 / (100 - P(22))
  Case 6                                ' fixed courses & width
    P(21) = 100 * (P(7) - P(11)) / P(7)
    P(22) = 100 * (P(14) - P(10)) / P(14)
    P(12) = P(8) * (100 - P(22)) / 100
    P(13) = P(9) * (100 - P(22)) * (100 - P(21)) / 10000
End Select

'....................................convert as del to user units
Select Case TarUnit
  Case 0
    GoSub FinCoursesUser
    GoSub FinWalesUser
```

```
        GoSub FinWeightUser
        GoSub FinWidthUser
      Case 1
        GoSub FinWeightUser
        GoSub FinWidthUser
      Case 2
        GoSub FinCoursesUser
        GoSub FinWalesUser
      Case 3
        GoSub FinWalesUser
        GoSub FinWidthUser
      Case 4
        GoSub FinCoursesUser
        GoSub FinWalesUser
        GoSub FinWeightUser
      Case 5
        GoSub FinCoursesUser
        GoSub FinWalesUser
        GoSub FinWidthUser
      Case 6
        GoSub FinWalesUser
        GoSub FinWeightUser
    End Select

    GoSub FinYieldUser

Return


FinCoursesUser:
    ' --------------------------- (Fin Courses in user units)
    If CrsUnit = 0 Then x1 = 1
    If CrsUnit = 1 Then x1 = 3
    If CrsUnit = 2 Then x1 = 10
    If CrsUnit = 3 Then x1 = 2.54
    If CrsUnit = 4 Then x1 = 3 / CperCell
    If CrsUnit = 5 Then x1 = 2.54 / CperCell
    P(17) = P(11) * x1

Return


FinWalesUser:
    ' ---------------------------- (Fin Wales in user units)
    If CrsUnit = 0 Then x1 = 1
    If CrsUnit = 1 Then x1 = 3
    If CrsUnit = 2 Then x1 = 10
    If CrsUnit = 3 Then x1 = 2.54
    If CrsUnit = 4 Then x1 = 3
    If CrsUnit = 5 Then x1 = 2.54
    P(18) = P(12) * x1
```

```
Return


FinWeightUser:
    ' ---------------------------- (Fin Weight in user units)
    If WtUnit = 0 Then P(19) = P(13)          'gsm
    If WtUnit = 1 Then P(19) = P(13) * GSMconverter  'osy

Return


FinYieldUser:
    ' ---------------------------- (Fin Yield in user units)
    'P(13) is the as delivered weight in gsm
    'P(14) is the open width in cm, untrimmed
    'NB allow for trimming
     Select Case WidUnit
       Case 0 To 3
         x3 = P(14)
       Case 4, 5
         x3 = P(14) - CurTrim
    End Select
    x1 = x3 / 100
    x2 = x1 * MTRconverter
    If YldUnit = 0 Then P(53) = P(13) * x1                  'g/m
    If YldUnit = 1 Then P(53) = P(13) * GSMconverter * x2   'o/y
    If YldUnit = 2 Then P(53) = P(13) * x1 * 0.9144         'g/y
    If YldUnit = 3 Then P(53) = 1000 / P(13) / x1           'm/kg
    If YldUnit = 4 Then P(53) = 33.91 / P(13) / x2 * 16     'y/lb
    If YldUnit = 5 Then P(53) = 1000 / P(13)                'sm/kg
    If YldUnit = 6 Then P(53) = 33.91 / P(13) * 16          'sy/lb

Return


FinWidthUser:
    ' ---------------------------- (Fin Width in user units)
                                     'default units are cm open
    'NB CurTrim is in cm
    x1 = P(14)  'Ref width, cm open
    If WidUnit = 0 Then P(20) = x1 / 2                'cm tubular
    If WidUnit = 1 Then P(20) = x1 / 2.54 / 2         'inch tubular
    If WidUnit = 2 Then P(20) = x1                    'cm open
    If WidUnit = 3 Then P(20) = x1 / 2.54             'inch open
    If WidUnit = 4 Then P(20) = x1 - CurTrim    'cm op trim
    If WidUnit = 5 Then P(20) = (x1 - CurTrim) / 2.54   'in op trim

Return


FinCoursesDefault:
    ' ------------------------ (Fin Courses in default units)
```

```
    If CrsUnit = 0 Then x1 = 1
    If CrsUnit = 1 Then x1 = 3
    If CrsUnit = 2 Then x1 = 10
    If CrsUnit = 3 Then x1 = 2.54
    If CrsUnit = 4 Then x1 = 3 / CperCell
    If CrsUnit = 5 Then x1 = 2.54 / CperCell
    P(11) = P(17) / x1

Return


FinWalesDefault:
    ' ------------------------- (Fin Wales in default units)
    If CrsUnit = 0 Then x1 = 1
    If CrsUnit = 1 Or CrsUnit = 4 Then x1 = 3
    If CrsUnit = 2 Then x1 = 10
    If CrsUnit = 3 Or CrsUnit = 5 Then x1 = 2.54
    P(12) = P(18) / x1

Return


FinWeightDefault:
    ' ------------------------- (Fin weight in default units)
    'default units are gsm
    If WtUnit = 0 Then P(13) = P(19)                    'gsm
    If WtUnit = 1 Then P(13) = P(19) / GSMconverter            'osy

Return


FinYieldDefault:
    ' ----------------- (Fin yield (weight) in default units)
    'default units are gsm
    'P(53) is the yield in user units
    'P(14) is the open width in cm, untrimmed

    x1 = P(14) / 100
    x2 = x1 * MTRconverter
    If YldUnit = 0 Then P(52) = P(53) * x1                 'g/m
    If YldUnit = 1 Then P(52) = P(53) / x2 / GSMconverter         'o/y
    If YldUnit = 2 Then P(52) = P(53) / x1 * MTRconverter        'g/y
    If YldUnit = 3 Then P(52) = 1000 / P(53) / x1          'm/kg
    If YldUnit = 4 Then P(52) = 16 / P(53) / x2 / GSMconverter    'y/lb
    If YldUnit = 5 Then P(52) = 1000 / P(53)               'sm/kg
    If YldUnit = 6 Then P(52) = 16 / P(53) / GSMconverter         'sy/lb

Return


FinWidthDefault:
    ' ------------------------- (Fin width in default units)
```

```
                                    Engine
                                'default units are cm open
    'NB CurTrim is in cm
    x1 = P(20)   'Fin Width, user
    If WidUnit = 0 Then P(14) = x1 * 2                      'cm tubular
    If WidUnit = 1 Then P(14) = x1 * 2.54 * 2               'inch tubular
    If WidUnit = 2 Then P(14) = x1                          'cm open
    If WidUnit = 3 Then P(14) = x1 * 2.54                   'inch open
    If WidUnit = 4 Then P(14) = x1 + CurTrim          'cm open trimmed
    If WidUnit = 5 Then P(14) = x1 * 2.54 + CurTrim   'inch open trimmed

Return


End Sub



'.....................................................................................
...........
Public Sub SetCurrentInputs(ByVal Ct As Integer, ByVal M As Integer, MisMach As
Boolean)

 'returns the inputs for the Starfish engine
      'Inputs(1) is first finishing target
      'Inputs(2) is second finishing target
      'Inputs(3) is yarn count 1
      'Inputs(4) is SL 1
'      'Inputs(5) is inlay count when FabTyp = 6
      'Inputs(5) is inlay count when FabTyp = Fab2tf
      'Inputs(6) is SL 2
      'Inputs(7) is number of Needles
      'Inputs(8) is Targets Unit for this calculation
      'Inputs(9) is Run-in Ratio

    Dim x1 As Single, x2 As Single
    Dim MisTarg As Boolean
    Dim Msg As String, Title As String
    Dim i As Long

    'Props array subscripts for Targets
    'should not be needed here but do it anyway
    Select Case TarUnit
      Case 0    'L Shr & W Shr
        Tar1 = 7
        Tar2 = 8
      Case 1    'Crses & Wales
        Tar1 = 2
        Tar2 = 3
      Case 2    'Wt & Wid
        Tar1 = 4
        Tar2 = 5
      Case 3    'Wt & Crses
```

```
    Tar1 = 4
    Tar2 = 2
  Case 4    'Lshr & Wid
    Tar1 = 7
    Tar2 = 5
  Case 5    'Wt & Lshr
    Tar1 = 4
    Tar2 = 7
  Case 6    'Crses & Wid
    Tar1 = 2
    Tar2 = 5
End Select

MisTarg = False
MisMach = False

For i = 1 To 7
  If Mac(M, i) = 0 Then MisMach = True
Next
If MisMach Then
  Msg = "Missing Machine Data" & NewLines2
  Msg = Msg & "Please check your machine selections"
  Title = " Missing Data"
  MsgBox Msg, BtnOK, Title
  Exit Sub
End If

Inputs(1) = FinFab(Tar1 - 1, Ct, M) 'target 1
Inputs(2) = FinFab(Tar2 - 1, Ct, M) 'target 2

'check for missing targets - only shrinkage can be zero
If Tar1 <> 7 And Tar1 <> 8 Then
  If Inputs(1) = 0 Then MisTarg = True
End If
If Tar2 <> 7 And Tar2 <> 8 Then
  If Inputs(2) = 0 Then MisTarg = True
End If

'check for missing yarn count
If CurTexVals(1, Ct) = 0 Then
  CurTexVals(1, Ct) = DefCounts(FabTyp, Ct)
  If CurTexVals(1, 0) < Ct Then
    CurTexVals(1, 0) = Ct
  End If
End If
'belt & braces
If CurTexVals(1, Ct) = 0 Then
  CurTexVals(1, Ct) = StdTex(FabTyp)
  If CurTexVals(1, 0) = 0 Then
    CurTexVals(1, 0) = 1
  ElseIf CurTexVals(1, 0) < Ct Then
    CurTexVals(1, 0) = Ct
```

```
  End If
End If

'face count, grey, user
If QualSpec(1, Ct, M) = 0 Then
  x1 = CurTexVals(1, Ct)
  If CountSys = 1 Then x1 = 590.54 / x1
  If CountSys = 2 Then x1 = 1000 / x1
  QualSpec(1, Ct, M) = x1
End If
Inputs(3) = QualSpec(1, Ct, M)

'allow for missing SL
If QualSpec(2, Ct, M) = 0 Then
  x1 = CurSLcmVals(1, Ct, M)
  If x1 = 0 Then
    'DefTF always in cgs units
    x2 = DefTF(FabTyp)
    'belt & braces
    If x2 < MinTF(FabTyp) Or x2 > MaxTF(FabTyp) Then
      Select Case FabTyp
        Case FabInt
          x2 = 13
        Case Fab2tf
          x2 = 15
        Case Else
          x2 = 16
      End Select
    End If
    x1 = Sqr(CurTexVals(1, Ct)) / x2
    CurSLcmVals(1, Ct, M) = x1
  End If
  'convert SL to user units
  x1 = ConvertSL(x1, SLcm, TiUnit)
  QualSpec(2, Ct, M) = x1
End If
Inputs(4) = QualSpec(2, Ct, M)      'face SL, grey, user

'allow for missing Inlay count
If FabTyp = Fab2tf Then
  x1 = QualSpec(5, Ct, M)
  If x1 = 0 Then
    x1 = CurTexVals(3, Ct)
    If x1 = 0 Then x1 = CurTexVals(1, Ct)
    If CountSys = 0 Then QualSpec(5, Ct, M) = x1
    If CountSys = 1 Then QualSpec(5, Ct, M) = 590.54 / x1
    If CountSys = 3 Then QualSpec(5, Ct, M) = 1000 / x1
  End If
End If
Inputs(5) = QualSpec(5, Ct, M)     'inlay yarn count if fleece

'allow for missing inlay SL
```

```
    If QualSpec(6, Ct, M) = 0 And FabTyp = Fab2tf Then
      x1 = CurSLcmVals(3, Ct, M)
      If x1 = 0 Then x1 = DefTF(NumFabrics) / Mac(M, 1) * 2.54
      CurSLcmVals(3, Ct, M) = x1
      'convert SL to user units
      x1 = ConvertSL(x1, SLcm, TiUnit)
      QualSpec(6, Ct, M) = x1
    End If
    Inputs(6) = QualSpec(6, Ct, M)     'SL, yarn 2

    'allow for missing needles
    If QualSpec(0, Ct, M) = 0 Then QualSpec(0, Ct, M) = Mac(M, 3)

'This gives problems if QualSpec(0, , ) still contains old values
'from a previous machine
    Inputs(7) = QualSpec(0, Ct, M)      'Needles

'But this gives problems if the machine needles in QualSpec(0...)
'have been purposely altered
'    Inputs(7) = Mac(M, 3)      'Needles

    'allow for missing targets
    Inputs(8) = TarUnit
    If MisTarg Then
      MsgBox "Missing Targets - Using Default Shrinkages"
      Inputs(8) = 0
      Inputs(1) = DefShr(FabTyp, 0)
      Inputs(2) = DefShr(FabTyp, 1)
    End If

    'allow for missing RunRat
'    If QualSpec(8, Ct, M) = 0 And (FabTyp = 4 Or FabTyp = 5) Then
    If QualSpec(8, Ct, M) = 0 And (FabTyp = FabSxt Or FabTyp = FabXt6) Then
      QualSpec(8, Ct, M) = 1
    End If
    Inputs(9) = QualSpec(8, Ct, M)     'RunRat if crosstuck

    'check for missing ProdInputs values
    'NB Default RollWt & LotWt always in Kg

    'Rpm
    If ProdInputs(1, Ct, M) = 0 Then ProdInputs(1, Ct, M) = Mac(M, 5)

    'Speed Factor
    If ProdInputs(2, Ct, M) = 0 Then ProdInputs(2, Ct, M) = Mac(M, 6)

    'Feeders
    If ProdInputs(3, Ct, M) = 0 Then ProdInputs(3, Ct, M) = Mac(M, 4)

    'Efficiency %
    If ProdInputs(4, Ct, M) = 0 Then
      If DefKnit(FabTyp, 1) = 0 Then DefKnit(FabTyp, 1) = 95
```

```
      ProdInputs(4, Ct, M) = DefKnit(FabTyp, 1)
    End If

    'Roll Wt, Kg
    If ProdInputs(5, Ct, M) = 0 Then
      If DefKnit(FabTyp, 0) = 0 Then DefKnit(FabTyp, 0) = 25
      ProdInputs(5, Ct, M) = DefKnit(FabTyp, 0)
      If WtUnit = 1 Then
        'convert to imperial
        ProdInputs(5, Ct, M) = ProdInputs(5, Ct, M) * KGconverter
      End If
    End If

    'Default Lot Wt, Kg
    If ProdInputs(7, Ct, M) = 0 Then
      If DefKnit(FabTyp, 2) = 0 Then DefKnit(FabTyp, 2) = 1
      ProdInputs(7, Ct, M) = DefKnit(FabTyp, 2)
      If WtUnit = 1 Then
        'convert to imperial
        ProdInputs(7, Ct, M) = ProdInputs(7, Ct, M) * KGconverter
      End If
    End If

    'Yarn waste %, default
    If ProdInputs(8, Ct, M) < 0 Or ProdInputs(8, Ct, M) > 25 Then
      ProdInputs(8, Ct, M) = DefKnit(FabTyp, 3)
    End If

    'Shift time
    If CurShift = 0 Then
      If DefKnit(FabTyp, 4) = 0 Then DefKnit(FabTyp, 4) = 8
      CurShift = DefKnit(FabTyp, 4)
    End If

End Sub

'.....................................................................
.
Public Sub SetDefaultInputs(ByVal Ct As Integer, ByVal M As Integer)

'NB no check for missing machine data
'NB: changed to access FinFab() rather than Props()

    Dim Ndls As Single, Diam As Single, Gge As Single
    Dim x1 As Single, x2 As Single
    Dim TF As Single

    Ndls = Mac(M, 3)
    Diam = Mac(M, 2)
    Gge = Mac(M, 1)
    QualSpec(0, Ct, M) = Ndls
```

```
'NB Default RollWt & LotWt always in Kg

'Rpm
ProdInputs(1, Ct, M) = Mac(M, 5)

'Speed Factor
ProdInputs(2, Ct, M) = Mac(M, 6)

'Feeders
ProdInputs(3, Ct, M) = Mac(M, 4)

'Default Efficiency %
ProdInputs(4, Ct, M) = DefKnit(FabTyp, 1)

'Default Roll Wt, Kg
ProdInputs(5, Ct, M) = DefKnit(FabTyp, 0)

'Default Lot Wt, Kg
ProdInputs(7, Ct, M) = DefKnit(FabTyp, 2)

'Convert RollWt & Lot Wt to Imperial
If WtUnit = 1 Then
  ProdInputs(5, Ct, M) = ProdInputs(5, Ct, M) * KGconverter
  ProdInputs(7, Ct, M) = ProdInputs(7, Ct, M) * KGconverter
End If

'Yarn waste %, default
ProdInputs(8, Ct, M) = DefKnit(FabTyp, 3)

'Shift time
CurShift = DefKnit(FabTyp, 4)

' temp targets, shrinkage
If FinFab(6, Ct, M) < 0 Or FinFab(6, Ct, M) > 15 Then
  FinFab(6, Ct, M) = DefShr(FabTyp, 0)
End If
If FinFab(7, Ct, M) < 0 Or FinFab(7, Ct, M) > 15 Then
  FinFab(7, Ct, M) = DefShr(FabTyp, 1)
End If
Inputs(1) = FinFab(6, Ct, M)
Inputs(2) = FinFab(7, Ct, M)

'face count, tex
x1 = CurTexVals(1, Ct)
If x1 = 0 Then
  x1 = StdTex(FabTyp)
Else
  If CountSys = 1 Then x1 = 590.54 / x1
  If CountSys = 2 Then x1 = 1000 / x1
End If
Inputs(3) = x1
```

```
'check whether SL1 has been set
If CurSLcmVals(1, Ct, M) > 0 Then
  x1 = CurTexVals(1, Ct)        'tex
  x2 = CurSLcmVals(1, Ct, M)    'SL, cm
  TF = Sqr(x1) / x2
  'make sure SL is reasonable
  If TF > NormTF(FabTyp, 1) Then TF = DefTF(FabTyp)
  If TF < NormTF(FabTyp, 0) Then TF = DefTF(FabTyp)
  x2 = Sqr(x1) / TF
Else
  'calculate SL1 from Default TF
  'DefTF always in cgs units - no need to convert
  TF = DefTF(FabTyp)
  x1 = CurTexVals(1, Ct)
  x2 = Sqr(x1) / TF      'SL in cm
End If
CurSLcmVals(1, Ct, M) = x2
'convert SL to user units
Inputs(4) = ConvertSL(x2, SLcm, TiUnit)

'check if inlay count has been set
If CurTexVals(3, Ct) > 0 Then
  If CountSys = 0 Then
    Inputs(5) = CurTexVals(3, Ct)
  ElseIf CountSys = 1 Then
    Inputs(5) = 590.54 / CurTexVals(3, Ct)
  Else
    Inputs(5) = 1000 / CurTexVals(3, Ct)
  End If
Else
  'set inlay count same as face and put into CurTexVals as tex
  Inputs(5) = Inputs(3)
  If CountSys = 0 Then
    CurTexVals(3, Ct) = Inputs(5)
  ElseIf CountSys = 1 Then
    CurTexVals(3, Ct) = 590.54 / Inputs(5)
  Else
    CurTexVals(3, Ct) = 1000 / Inputs(5)
  End If
End If

'check if Inlay SL has been set
If CurSLcmVals(3, Ct, M) > 0 Then
  x1 = CurSLcmVals(3, Ct, M)
Else
  'calculate & put into CurSLcmVals
  x1 = DefTF(NumFabrics) / Gge * 2.54
  CurSLcmVals(3, Ct, M) = x1
End If
'convert SL to user units
Inputs(6) = ConvertSL(x1, SLcm, TiUnit)
```

```
    Inputs(7) = Ndls
    Inputs(8) = 0       'Targets are shrinkages
    Inputs(9) = 1       'Run in Ratio

End Sub



Sub SetFabric()

'   stuff to do when change fabric type

    Sc = ScSw(FabTyp, 0)
    Sw = ScSw(FabTyp, 1)
    Grey(2) = GreySLRat
    Grey(3) = 0.967             '.967 is grey WtLoss
    Grey(4) = -3.3
    Grey(1) = Grey(3) / Grey(2)
    Grey(10) = Grey(1)
    Grey(11) = Grey(2)
    Grey(12) = Grey(1)
    Grey(13) = Grey(2)
    Grey(14) = Grey(3)
    Grey(15) = Grey(3)

    Select Case FabTyp
      Case FabInt, FabRib1, FabRib2, FabPsj, Fab2tf
        CperCell = 1
      Case FabDxt, FabSxt
        CperCell = 4
      Case FabXt6
        CperCell = 6
    End Select

    If Not CurMachFile < 0 Then
      DefSortOpt(CurMachFile) = SortOpt
    End If
    Select Case FabTyp
      Case FabInt
        CurMachFile = IntMachs
      Case FabRib1, FabRib2
        CurMachFile = RibMachs
      Case Else
        CurMachFile = SjMachs
    End Select
    SortOpt = DefSortOpt(CurMachFile)


End Sub
```

```
'.....................................................
Sub SetProcess()

'assigns initial values to Eq()
'NB Eq(3) comes from SetShade and is needed here
'so SetShade must have been done beforehand, if necessary
'Eq(4) is also set by SetShade but is not needed here
'Stop

    If ProcTyp < NumProcs Then
      ProcName = frmProcTyp.lstProcTyp.List(ProcTyp)
      ProcName = RTrim(ProcName)
      Eq(5) = ProcFc(FabTyp, ProcTyp)
      Eq(6) = ProcFw(FabTyp, ProcTyp)
      Eq(2) = SLRat(ProcTyp)
      Eq(1) = Eq(3) / Eq(2)
      Eq(10) = Eq(1)
      Eq(11) = Eq(2)
      Eq(12) = Eq(1)
      Eq(13) = Eq(2)
      Eq(14) = Eq(3)
      Eq(15) = Eq(3)
      ConvertEq5andEq6 0, YarnTyp
    Else
      ConvertEq5andEq6 Eq(9), YarnTyp
    End If
    Fc = Eq(5)
    Fw = Eq(6)
    Eq(9) = YarnTyp

    GetStdProcCalibRatio

End Sub

'.....................................................
Sub SetShade(ByVal ShadeSetMode As Integer)

'sets value for Eq(3) - weight loss in processing
'SetMode = 0 for complete reset, using frmShade.lstShade
'SetMode = 1 to merely update ShadeCal, eg after Calibration
'            using Eq(4), without changing Eq(3)
'NB for Mode 0, ProcTyp must have been set

    Dim strZ As String, Sname As String
    Dim StdCal As Variant
    Dim i As Integer

    Select Case ShadeSetMode
      Case 0        'complete reset
        ShadeName = ShadeNames(Shade)
        strZ = WtLossNames(Shade)
        StdCal = GetVal(strZ)
```

```
'        StdCal = Val(strZ)
        ShadeCal = Format(StdCal, "+#0.00; -#0.00")
        Eq(4) = StdCal
        Eq(3) = (100 + Eq(4)) / 100

        'modify Eq(1) if necessary
        If ProcTyp < NumProcs Then
          Eq(2) = SLRat(ProcTyp)
          Eq(1) = Eq(3) / Eq(2)
          Eq(10) = Eq(1)
          Eq(12) = Eq(1)
        End If

      Case 1        'reset ShadeCal only
'        ShadeCal = "[ " & Format(Eq(4), "+#0.00; -#0.00") & " ]"
        ShadeCal = Format(Eq(4), "+#0.00; -#0.00")
'        If Eq(4) <> StdCal Then ShadeName = " Udp       "
        If ShadeName = "" Then ShadeName = "Udp"

    End Select

End Sub

'.....................................................................
Sub SetYarnType()

    Dim Ix As Integer

    Select Case YarnTyp
      Case 0
        Grey(5) = GreyFcFw(FabTyp, 0)      'combed ring
        Grey(6) = GreyFcFw(FabTyp, 1)
      Case 1
        Grey(5) = GreyFcFw(FabTyp, 0) * 1.04  'carded ring
        Grey(6) = GreyFcFw(FabTyp, 1) * 0.96
      Case 2
        Grey(5) = GreyFcFw(FabTyp, 0) * 0.9   'carded rotor
        Grey(6) = GreyFcFw(FabTyp, 1) * 0.9
      Case 3
        Grey(5) = GreyFcFw(FabTyp, 0) * 1.2       'two-fold
        Grey(6) = GreyFcFw(FabTyp, 1) * 0.93
    End Select

    If Eq(9) <> YarnTyp Then          'change of yarn type
      Ix = Eq(9)
      ConvertEq5andEq6 Ix, YarnTyp
      Fc = Eq(5)
      Fw = Eq(6)
    End If

    Eq(9) = YarnTyp
```

```
End Sub

'.....................................................................
Sub UpdateCalRat()
'NB does this always come immediately after Sub GetStdProcCalibRatio?
'answer is NO!

    Dim i As Integer
    Dim x1 As Variant, x2 As Variant

    x1 = Format(Eq(7) * 100, "###.0")
    x2 = Format(Eq(8) * 100, "###.0")
'    ProcCal = " [ " & x1 & " : " & x2 & " ]"
    ProcCal = x1 & " : " & x2

End Sub

'.....................................................................
.
Public Function GetShadeName(ByVal ShadeNum As Integer) As String

    Dim Sname As String, Wloss As String
    Dim Wid As Integer

    Sname = ShadeNames(ShadeNum)
    Wloss = "[ " & WtLossNames(ShadeNum) & " ]"
    frmShade.Font.Bold = True
    Wid = frmShade.TextWidth("Medium +   ")

    Do While frmShade.TextWidth(Sname) < Wid
      Sname = Sname & " "
    Loop

    GetShadeName = Sname & Wloss

End Function

'.....................................................................
Public Function GetUdpName(F As Form, ByVal Z As String, X() As Single) As
String

' z is the process name
'x(0) is  courses calibration ratio - Eq(7)
'x(1) is the wales calibration ratio - Eq(8)
'x(2) is the WtRat equivalent - Eq(4)

    Dim Proc As String
    Dim Ccal As String
    Dim Wcal As String
    Dim WtCal As String
    Dim Wid As Integer
```

```
'    F.Font.Bold = True
    Wid = F.TextWidth("Wxxxxxxxxxxxxxxxxxxxxxxxxxxxxw")

    Proc = LTrim(RTrim(Z))
    Do While F.TextWidth(Proc) < Wid
      Proc = Proc & " "
    Loop
    Proc = Proc & Chr(9)
    Ccal = Format$(100 * X(0), "\ ###.0")
    Wcal = Format$(100 * X(1), "\ ###.0")
    Proc = Proc & "[ " & Ccal & " :" & Wcal & " ]"
    WtCal = Format$(X(2), "+#0.00; -#0.00")
    Proc = Proc & Chr(9) & "[ " & WtCal & " ]"

    GetUdpName = Proc

End Function


'.....................................................................
Public Sub GetFinFabLimits(ByVal Ct As Integer, ByVal M As Integer)
    'Establishes Max and Min Values for
    'Data Entry for current quality
    'and places them in FinFabLimits array
    'NB has to be able to track current Ref Dims
    'Ct = CurQual
    'M = CurMach

    Dim Crses As Single, Wales As Single
    Dim Wt As Single, Wid As Single
    Dim x1 As Single, x2 As Single, x3 As Single


    Crses = Props(15, Ct, M) 'Current Ref Courses
    Wales = Props(16, Ct, M) 'Current Ref Wales
    Wt = Props(17, Ct, M)    'Current Ref Weight
    Wid = Props(18, Ct, M)   'Current Ref Width

    'Courses & Wales
    FinFabLimits(1, 0) = Crses * 0.75
    FinFabLimits(1, 1) = Crses * 1.05
    FinFabLimits(2, 0) = Wales * 0.75
    FinFabLimits(2, 1) = Wales * 1.05

    'weight limits depend on Wmode and existing shrinkages
    'in order to stop shrinkages exceeding limits
    x1 = (100 - FinFab(7, CurQual, 0)) / 100
    If Wmode Then
      'depends on width shrinkage only
      FinFabLimits(3, 0) = Wt * 0.751 * x1
      FinFabLimits(3, 1) = Wt * 1.049 * x1
    Else
```

```
      'depends on the one nearest the limit
      x2 = (100 - FinFab(6, CurQual, 0)) / 100
      'lightest weight - highest shrinkage
      If x1 < x2 Then x3 = x2 Else x3 = x1
      FinFabLimits(3, 0) = Wt * 0.751 * x3
      'heaviest weight - lowest shrinkage
      If x1 < x2 Then x3 = x1 Else x3 = x2
      FinFabLimits(3, 1) = Wt * 1.049 * x3
    End If

    'Width
'    FinFabLimits(4, 0) = Wid * 0.95
'    FinFabLimits(4, 1) = Wid * 1.25
    FinFabLimits(4, 0) = Wid / 1.05
    FinFabLimits(4, 1) = Wid / 0.75

    '5 is empty (yield)

    'Shrinkages
    FinFabLimits(6, 0) = -5
    FinFabLimits(6, 1) = 25
    FinFabLimits(7, 0) = -5
    FinFabLimits(7, 1) = 25

End Sub

'.....................................................................
Public Sub GetQualSpecLimits(ByVal Ct As Integer, ByVal M As Integer)
    'Establishes Max and Min Values for
    'Data Entry for current quality
    'and places them in QualSpecLimits array
    'NB has to be able to track current Ref Dims
    'Ct = CurQual
    'M = CurMach

    Dim MaxSL As Single, MinSL As Single
    Dim MaxInSL As Single, MinInSL As Single
    Dim MaxCL As Single, MinCL As Single
    Dim MaxInCL As Single, MinInCL As Single
    Dim x1 As Single, x2 As Single, x3 As Single

    Ct = CurQual
    M = CurMach

    x1 = CurTexVals(1, Ct)              'count in tex
    MinSL = Sqr(x1) / MaxTF(FabTyp)         'SL in cm
    MaxSL = Sqr(x1) / MinTF(FabTyp)
    'Min Inlay SL is cylinder circumference * MinInlayTF
    MinInSL = PI * Mac(CurMach, 2) / Mac(CurMach, 3) * 2.54 * MinInlayTF
    MaxInSL = PI * Mac(CurMach, 2) / Mac(CurMach, 3) * 2.54 * MaxInlayTF
    'CL = SL * Needles
    MinCL = MinSL * QualSpec(0, Ct, M)
```

```
    MaxCL = MaxSL * QualSpec(0, Ct, M)
    'double up for 1x1 rib
    If FabTyp = FabRib1 Then
      MaxCL = MaxCL * 2
      MinCL = MinCL * 2
    End If
    'double up for 2x2 rib
'    If FabTyp = 2 Then
    If FabTyp = FabRib2 Then
      MaxCL = MaxCL * 2
      MinCL = MinCL * 2
    End If
    MinInCL = MinInSL * QualSpec(0, Ct, M)
    MaxInCL = MaxInSL * QualSpec(0, Ct, M)
    'convert SL to user
    MinSL = ConvertSL(MinSL, SLcm, TiUnit)
    MaxSL = ConvertSL(MaxSL, SLcm, TiUnit)
    MinInSL = ConvertSL(MinInSL, SLcm, TiUnit)
    MaxInSL = ConvertSL(MaxInSL, SLcm, TiUnit)
    'convert CL to inches, if necessary
    If TiUnit = SLin Then
      MinCL = MinCL / 2.54
      MaxCL = MaxCL / 2.54
      MinInCL = MinInCL / 2.54
      MaxInCL = MaxInCL / 2.54
    End If

    'Needles
    If Not FabTyp = FabRib2 Then
      QualSpecLimits(0, 0) = MaxMin(23, 1)
      QualSpecLimits(0, 1) = MaxMin(23, 0)
    Else
      QualSpecLimits(0, 0) = MaxMin(23, 1) * RibSetOut
      QualSpecLimits(0, 1) = MaxMin(23, 0) * RibSetOut
    End If

    'Face Count
    QualSpecLimits(1, 0) = MaxMin(9 + CountSys, 1)
    QualSpecLimits(1, 1) = MaxMin(9 + CountSys, 0)

    'Yarn1 StLen & CL & TF
    QualSpecLimits(2, 0) = MinSL
    QualSpecLimits(2, 1) = MaxSL
    QualSpecLimits(3, 0) = MinCL
    QualSpecLimits(3, 1) = MaxCL
    QualSpecLimits(4, 0) = MinUserTF
    QualSpecLimits(4, 1) = MaxUserTF

    'Yarn2 Count, StLen, CL & RunRat / TF
    Select Case FabTyp
'      Case 4, 5
      Case FabSxt, FabXt6
```

```
        QualSpecLimits(5, 0) = MaxMin(9 + CountSys, 1)
        QualSpecLimits(5, 1) = MaxMin(9 + CountSys, 0)
        QualSpecLimits(6, 0) = MinSL * 0.8
        QualSpecLimits(6, 1) = MaxSL * 1.2
        QualSpecLimits(7, 0) = MinCL * 0.8
        QualSpecLimits(7, 1) = MaxCL * 1.2
        QualSpecLimits(8, 0) = 0.8
        QualSpecLimits(8, 1) = 1.2
'      Case 6
      Case Fab2tf
        QualSpecLimits(5, 0) = MaxMin(9 + CountSys, 1)
        QualSpecLimits(5, 1) = MaxMin(9 + CountSys, 0)
        QualSpecLimits(6, 0) = MinInSL
        QualSpecLimits(6, 1) = MaxInSL
        QualSpecLimits(7, 0) = MinInCL
        QualSpecLimits(7, 1) = MaxInCL
        QualSpecLimits(8, 0) = MinInlayTF
        QualSpecLimits(8, 1) = MaxInlayTF
    End Select

End Sub
```